**Dawson College**

**Electrical Engineering Technology Department**

**Introduction to Internet of Things**

**Project Name:**

Keypad Password

**Team Members:**

Jose Nunes (ID: 2436299), Adrian Lacraru (ID: 2433360)

**2. Project description**

Every house has a front door lock, but sometimes that isn’t enough. This project allows you to add a PIN lock anywhere. Although our project doesn’t include any motors, it only has an LCD display and a Keypad, it would be very simple to implement a motor of sorts to allow the pin to actually work as a locking/unlocking mechanism for drawers, cabinets or even safes.

This project promotes extra privacy and security over your goods that you want to keep extra safe.

**How it works:**

* The **Keypad** is an input device used to **input the PIN.**

-It features **numbers 0 through 9**, which can be used for the **PIN**.

-A **“#”** which is used as the **“enter”** key, to check if you imputed PIN is **correct or not**.

-An **“\*”** key which can be used to **retype** the password **without checking** it, in case you made a mistake and know about it and would like to try again without risking getting locked out.

-(The Keypad used also has “A”, “B”, “C”, and “D” keys, but they weren’t utilized in this project).

* The **LCD display** is an output device which is simply used to **give instructions and** **inform you** if you entered the **correct password or not**.

**Final assembly diagram:**

A diagram of a calculator

AI-generated content may be incorrect.

*Figure 1 – Project Diagram*

**3. Circuit Diagram**

**Input:**

* Keypad (Membrane Switch)

**Output:**

* LCD display (LCD 1602 Module)

Below is a description of the hardware used in this project, which is also shown graphically using Tinkercad.

|  |  |
| --- | --- |
| Part: | Arduino Uno Pin: |
| Keypad (Membrane Switch) | ROWS → A0, A1, A2, A3  COLS → D5, D4, D3, D2 |
| LCD display (LCD 1602 Module) | K → GND  A → VCC (+5V)  D7 → D12  D6 →D11  D5 →D10  D4 →D9  E → D8  RW → GND  RS →D7  V0 → GND (through a 1kΩ resistor)  VDD → VCC (+5V)  VSS → GND |
| +5v Rail | 5V Pin (red) |
| GND Rail | GND Pin (black) |

A close-up of a circuit board

AI-generated content may be incorrect.

*Figure 2 – Circuit Connections Diagram, image from Tinkercad.*

**Notes about the circuit assembly:**

1. We used the analog pins A0, A1, A2, A3, because we didn’t have enough digital pins due to the LCD display needing so many. It should work perfectly as intended despite the analog pins being used instead of the digital pins.

**4. Code Documentation**

**4.1 Libraries Used**

#include <Keypad.h>           // Includes the library for the keypad

#include <LiquidCrystal.h>    // Includes the library for the LCD display

Keypad library allows us to use functions designed for the keypad, allowing us to use simpler code and not have to code the entire keypad from scratch.

LiquidCrystal library, similarly, allows us to use built in functions to facilitate the code using the LCD display.

**4.2 Global Constant & Pins**

Below is the list of constant variables used in the code. Because they do not change during the code running, they can be used anywhere inside or outside other functions. Also, they can be used more than once in many blocks of code at a time.

|  |  |  |
| --- | --- | --- |
| Constant/Pin: | Purpose: | Type Value / pins: |
| Password | 4-digit password the user must enter correctly | “1234” |
| maxAttempts | Maximum # of failed attempts before lockout | 3 |
| ROWs | # of keypad rows | 4 |
| COLs | # of keypad columns | 4 |
| RowPins[] | Arduino pins connected to keypad rows | A0, A1, A2, A3 |
| ColPins[] | Arduino pins connected to keypad columns | 5, 4, 3, 2 |
| lcs | LCD projects are set up by using output pin inputs. | 7(rs), 8(en), 9(db4), 10(db5), 11(db6), 12(db7) |

**4.3 Function Descriptions and Responsibility**

|  |  |
| --- | --- |
| Functions | Descriptions & Responsibilities |
| Void setup() | * Set the LCD to 8 columns and 2 rows. * Clear any previous displays. * Prints “Enter Password:” on screen. |
| Void loop() | * Stops running when the system enters locked mode. * Reads keys when pressed from keypad. * Clear input is key “\*”. * Check password input is key “#”. * If digits are pressed (and input is less than 4), it sends a warning. * LCD masks the code with “\*” symbols. |
| Void clearInput() | * Empties the input (string). * Clear the LCD screen. * If the system isn’t in lock mode, the LCD will display “try again:” to allow user to retry. |
| Void checkPassword() | * Compare the user input to the correct password. * If correct = LCD prints “correct” and welcomes user. * If incorrect = LCD will display # of attempts, and “incorrect”. * If max attempts reached =locks the user out and displays locked message * Users will need to wait 24 hours (delay (86400000)) |

**4.4 Code Explanation:**

While our code can be found in our very own, [GitHub,](https://github.com/AdrianLacraru/Final-Project) we have explained how the code mainly works below.

1. **Keypad Reading – Continuous Polling**  
    Every cycle of loop() checks if a key is pressed on the 4x4 keypad using keypad.getKey().

* If a valid key is detected, it processes it immediately.
* The valid keys are digits (0–9), the clear key (\*), or the submit key (#).

1. **Password Input Buffer – Limited and Masked**  
    If a digit is pressed and the input is shorter than 4 characters:

* The digit is appended to the input string.
* A masked \* character is displayed on the second row of the LCD, maintaining secrecy.

1. **Input Reset – \* Button Logic**  
    Pressing \* clears the current input:

* clearInput() is called.
* The screen is cleared and displays "try again:", allowing the user to start over, without wasting an attempt.

1. **Password Validation – Triggered by # Key**  
    Pressing # initiates checkPassword() to compare the typed input with the stored password.

* If matched:
  + LCD displays "Correct" then "Welcome\Adrian & Jose".
  + Attempt counter resets.
* If incorrect:
  + The attempts counter is incremented.
  + LCD shows "Incorrect", and prompts retry (unless max attempts are reached, in which case it will lock out, ).

1. **Lockout Mechanism – After 3 Failed Attempts**  
    Once the attempts counter reaches 3:

* The system sets locked = true.
* LCD displays "LOCKED OUT" and "try again in 24h".
* Arduino pauses for 24 hours using delay(86400000), preventing further access.

1. **System Freeze – Locked Mode Disables Input**  
    When locked is true, the first line of loop() stops all logic from running:

* No keypad reads or responses occur during the lockout delay.

1. **Auto Reset – After Lockout Delay or Success**  
    After either:

* a successful password entry (resets attempts = 0), or
* A 24-hour lockout,  
   the system clears input, resets the screen, and returns to "Enter Password:" prompt for fresh interaction.

With all these steps being considered when creating a keypad password; it allows for password identification, secret typing, and full locking when all attempts are used up. Only when a full 24 hours are reached will a user be allowed to try again.

**5. Ethics, Privacy, or Security Disclaimer**

We have reflected on the ethical implications of our project. We see our project being used in every software application. This is due to the diverse ways we can use passwords to identify users and implement high security. This device/system is a security tool and with the code can be changed in almost any way. From encryption all the way to typing out characters on a keypad.

**6. References**

* Keypad tutorial: <https://www.youtube.com/watch?v=vl1-R6NsejM>
* Tinkercad: [Dashboard - Tinkercad](https://www.tinkercad.com/dashboard)
* Arduino Keypad library
* Arduino LCD display library